

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
Факультет компьютерных наук
Образовательная программа «Программная инженерия»**

СОГЛАСОВАНО

Научный руководитель, доцент
факультета компьютерных наук

_____ Янович Ю.А.
«__» _____ 2026 г.

УТВЕРЖДЕНО

Академический руководитель
образовательной программы
«Программная инженерия», старший
преподаватель департамента
программной инженерии

_____ Н. А. Павлов
«__» _____ 2026 г.

**КОНСТРУКТОР СМАРТ-КОНТРАКТОВ С ДИНАМИЧЕСКОЙ КОМИССИЕЙ
ДЛЯ ДЕЦЕНТРАЛИЗОВАННЫХ БИРЖ**

Техническое задание

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.01-01 ТЗ 01-1-ЛУ

Исполнители:

Студенты группы БПИ235

_____ / «Фролов И.Г.» /

«__» _____ 2026 г.

Инва.№ подп	Подп. и дата	Взам. инв.№	Инва.№ дубл.	Подп. и дата

УТВЕРЖДЕН

RU.17701729.05.01-01 ТЗ 01-1-ЛЮ

**КОНСТРУКТОР СМАРТ-КОНТРАКТОВ С ДИНАМИЧЕСКОЙ КОМИССИЕЙ
ДЛЯ ДЕЦЕНТРАЛИЗОВАННЫХ БИРЖ**

Текст программы

RU.17701729.05.01-01 ТЗ 01-1

Листов 10

Инов.№ подп	Подп. и дата	Взам. инв.№	Инов.№ дубл.	Подп. и дата

АННОТАЦИЯ

Техническое задание – это основной документ, оговаривающий набор требований и порядок создания программного продукта, в соответствии с которым производится разработка программы ее тестирование и приемка.

Настоящее Техническое задание направленное на разработку web-платформы, которая обрабатывает и токенизирует данные о здоровье пользователей содержит следующие разделы: «Введение», «Основания для разработки», «Назначение разработки», «Требования к программе», «Требования к программной документации», «Технико-экономические показатели», «Стадии и этапы разработки», «Порядок контроля и приемки», приложения [7].

В разделе «Введение» указано наименование и краткая характеристика области применения программы.

В разделе «Основания для разработки» указан документ, на основании которого ведется разработка, и наименование темы разработки.

В разделе «Назначение разработки» указано функциональное и эксплуатационное назначение создаваемого программного продукта.

Раздел «Требования к программе» содержит указание на основные требования к функциональным характеристикам программы, к её надежности и к условиям эксплуатации, к составу и параметрам технических средств, к информационной и программной совместимости, к маркировке и упаковке, к транспортировке и хранению, а также специальные требования.

Раздел «Требования к программным документам» содержит указание на предварительный состав программной документации и специальные требования к ней.

Раздел «Технико-экономические показатели» содержит информацию об ориентировочной экономической эффективности разработки, экономические преимущества разработки программы.

Раздел «Стадии и этапы разработки» содержит информацию о стадиях разработки, этапах и содержании работ.

В разделе «Порядок контроля и приемки» указаны общие требования к приемке работы.

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.101-77 [1]: Виды программ и программных документов.
2. ГОСТ 19.102-77 [2]: Стадии разработки.
3. ГОСТ 19.103-77 [3]: Обозначения программ и программных документов.
4. ГОСТ 19.104-78 [4]: Основные надписи.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

RU.17701729.05.01-01 ТЗ 01-1

5. ГОСТ 19.105-78 [5]: Общие требования к программным документам.
6. ГОСТ 19.106-78 [6]: Требования к программным документам, выполненным печатным способом.
7. ГОСТ 19.201-78 [7]: Техническое задание. Требования к содержанию и оформлению.

Изменения к данному Техническому заданию оформляются согласно ГОСТ 19.603-78 [12], ГОСТ 19.604-78 [13].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

СОДЕРЖАНИЕ

1. ТЕКСТ ПРОГРАММЫ	6
1.1. Исходный код программы	6
2. ОПИСАНИЕ ПРОГРАММЫ	7
2.1. Описание проекта	7
2.1.1. Основной функционал проекта включает:	7
2.1.1.1. Реализация библиотеки смарт-контрактов (хуков) динамических комиссий для Uniswap V4	7
2.1.1.2. Разработка нескольких типов хуков с различными алгоритмами расчета комиссий	7
2.1.1.3. Интеграция с оракулом Chainlink для получения актуальных ценовых данных . . .	7
2.1.1.4. Совместимость с ключевыми артефактами Uniswap V4	7
2.1.1.5. Инфраструктура для тестирования и развертывания	7
2.1.1.6. Веб-интерфейс для конструирования смарт-контрактов	7
2.1.1.7. Анализ эффективности алгоритмов на исторических данных	7
2.2. Структура проекта	7
2.2.1. Основные директории и файлы:	7
2.2.1.1. src/	7
2.2.1.2. script/	8
2.2.1.3. script/base/	8
2.2.1.4. test/	8
2.2.1.5. test/utills/	8
2.2.1.6. web_interface/	8
2.2.1.7. simulation/	8
2.2.1.8. lib/	8
2.2.1.9. foundry.toml	8
2.3. Описание работы компонентов	8
2.3.1. Смарт-контракты (хуки) на Solidity	8
2.3.1.1. MEVChargeHook.sol	8
2.3.1.2. PegStabilityHook.sol	8
2.3.1.3. Другие хуки (BAHook.sol, DAHook.sol, ABHook.sol)	8
2.3.1.4. Базовые контракты (BaseOverrideFee.sol)	9

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2.3.2. Функции контрактов:	9
2.3.2.1. getFee(...)	9
2.3.2.2. beforeSwap(...)	9
2.3.2.3. Интеграция с Chainlink Price Feeds	9
2.3.3. Скрипты развертывания (Solidity/Forge)	9
2.3.3.1. deployHook.s.sol	9
2.3.3.2. HookHelpers.sol	9
2.3.4. Тесты (Solidity/Forge)	9
2.3.4.1. MEVChargeHookFees.t.sol	9
2.3.4.2. testHook.t.sol	9
2.3.4.2.0.1. HookTest.sol	9
2.3.5. Веб-интерфейс (Python/Flask)	9
2.3.5.1. app.py	9
2.3.5.2. web_interface/templates/	10
2.3.6. Симуляция и анализ (Python)	10
2.3.6.1. simulation.py	10
2.3.6.2. fetch_binance_data.py	10
2.4. Технологии	10
2.4.1. Языки программирования	10
2.4.1.1. Solidity	10
2.4.1.2. Python	10
2.4.1.3. HTML/CSS/JavaScript	10
2.4.2. Инструменты разработки	10
2.4.2.1. Foundry (Forge, Anvil, Cast)	10
2.4.2.2. Uniswap V4 Core & Periphery	10
2.4.2.3. Chainlink	10
2.4.2.4. Binance API	10
2.4.2.5. OpenZeppelin Contracts & UniswapHooks	10
2.4.2.6. Flask	10

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1. ТЕКСТ ПРОГРАММЫ**1.1. Исходный код программы**

Исходный код программы доступен в удаленном репозитории GitHub: <https://github.com/ifde/dex>

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2. ОПИСАНИЕ ПРОГРАММЫ

2.1. Описание проекта

Данный проект реализует библиотеку смарт-контрактов (хуков) для протокола Uniswap V4, предназначенную для динамического изменения комиссий в пулах ликвидности децентрализованных бирж (DEX). Он позволяет автоматически корректировать размер комиссии на основе рыночных условий (волатильности, объема торгов, арбитражного спреда), что снижает непостоянные потери для поставщиков ликвидности и повышает эффективность DEX.

2.1.1. Основной функционал проекта включает:

2.1.1.1. Реализация библиотеки смарт-контрактов (хуков) динамических комиссий для Uniswap V4

Базовое наследование от контракта BaseOverrideFee с унифицированным интерфейсом.

2.1.1.2. Разработка нескольких типов хуков с различными алгоритмами расчета комиссий

На основе истории свопов, объема торгов, данных с централизованных бирж (CEX).

2.1.1.3. Интеграция с оракулом Chainlink для получения актуальных ценовых данных

Обеспечение совместимости с внешними источниками информации.

2.1.1.4. Совместимость с ключевыми артефактами Uniswap V4

PoolManager, SwapRouter, PositionManager для создания пулов, управления ликвидностью и выполнения свопов.

2.1.1.5. Инфраструктура для тестирования и развертывания

Скрипты для автоматического развертывания на локальном блокчейне (Anvil), тесты симуляции торговой активности.

2.1.1.6. Веб-интерфейс для конструирования смарт-контрактов

Выбор типа комиссии, настройка параметров, генерация кода и сборка готового проекта.

2.1.1.7. Анализ эффективности алгоритмов на исторических данных

Использование данных Binance API для тестирования и сравнения.

2.2. Структура проекта

2.2.1. Основные директории и файлы:

2.2.1.1. src/

Директория с основными смарт-контрактами хуков динамических комиссий (например, MEVChargeHook.sol, PegStabilityHook.sol и др.).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2.2.1.2. script/

Скрипты для развертывания хуков и создания пулов ликвидности (deployHook.s.sol, deployHookAndCreatePool.s.sol).

2.2.1.3. script/base/

Вспомогательные контракты и утилиты для развертывания (HookHelpers.sol, LiquidityHelpers.sol, BaseScript.sol).

2.2.1.4. test/

Тесты для смарт-контрактов, включая симуляцию торговой активности и изменение комиссий (MEVChargeHookFees.t.sol, testHook.t.sol).

2.2.1.5. test/utils/

Утилиты для тестирования (HookTest.sol, HookConstants.sol, HookFlags.sol).

2.2.1.6. web_interface/

Веб-интерфейс на Python/Flask для конфигурации и генерации хуков (app.py, templates/, static/).

2.2.1.7. simulation/

Скрипты для симуляции торговли и анализа эффективности на исторических данных (simulation.py, TRADE_SIMULATION.md).

2.2.1.8. lib/

Внешние зависимости: Uniswap V4 Core/Periphery, Chainlink, OpenZeppelin.

2.2.1.9. foundry.toml

Конфигурация Foundry (Forge).

2.3. Описание работы компонентов**2.3.1. Смарт-контракты (хуки) на Solidity**

Основные контракты реализуют логику динамических комиссий.

2.3.1.1. MEVChargeHook.sol

Хук для комиссий на основе MEV (Maximal Extractable Value) и арбитражного спреда.

2.3.1.2. PegStabilityHook.sol

Хук для стабилизации цены через корректировку комиссий на основе отклонения от целевой цены.

2.3.1.3. Другие хуки (BAHook.sol, DAHook.sol, ABHook.sol)

Различные алгоритмы расчета комиссий.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2.3.1.4. Базовые контракты (BaseOverrideFee.sol)

Унифицированный интерфейс для всех хуков.

2.3.2. Функции контрактов:**2.3.2.1. getFee(...)**

Расчет комиссии на основе параметров свопа и рыночных данных.

2.3.2.2. beforeSwap(...)

Хук, вызываемый перед свопом для корректировки комиссии.

2.3.2.3. Интеграция с Chainlink Price Feeds

Используется для получения внешних данных (цен токенов на централизованных биржах).

2.3.3. Скрипты развертывания (Solidity/Forge)

Foundry используется для компиляции, тестирования и развертывания на локальном (Anvil) или реальном блокчейне.

2.3.3.1. deployHook.s.sol

Развертывание выбранного хука с указанием адресов оракулов.

2.3.3.2. HookHelpers.sol

Утилиты для развертывания, включая создание mock-оракулов и токенов.

2.3.4. Тесты (Solidity/Forge)

Тесты симулируют работу хуков. Тесты включают fuzz-тестирование, симуляцию свопов и проверку корректности комиссий.

2.3.4.1. MEVChargeHookFees.t.sol

Тестирование комиссий и взаимодействия с Uniswap V4.

2.3.4.2. testHook.t.sol

Общие тесты для всех хуков.

2.3.4.2.0.1. HookTest.sol

Базовый контракт для тестирования с развертыванием инфраструктуры.

2.3.5. Веб-интерфейс (Python/Flask)

Интерфейс позволяет конструировать хуки. Функционал: выбор типа хука и параметров, генерация Solidity-кода, сборка и скачивание ZIP-проекта с зависимостями

2.3.5.1. app.py

Основной сервер на Flask, обрабатывающий запросы.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2.3.5.2. web_interface/templates/

HTML-шаблоны для интерфейса.

2.3.6. Симуляция и анализ (Python)**2.3.6.1. simulation.py**

Скрипт для симуляции торговли с использованием рыночных данных.

2.3.6.2. fetch_binance_data.py

Скрипт для получения актуальных рыночных данных с Binace через API

2.4. Технологии**2.4.1. Языки программирования****2.4.1.1. Solidity**

Основной язык для смарт-контрактов и хуков.

2.4.1.2. Python

Для веб-интерфейса, симуляции и анализа данных.

2.4.1.3. HTML/CSS/JavaScript

Для фронтенда веб-интерфейса.

2.4.2. Инструменты разработки**2.4.2.1. Foundry (Forge, Anvil, Cast)**

Фреймворк для разработки, тестирования и развертывания смарт-контрактов.

2.4.2.2. Uniswap V4 Core & Periphery

Базовые контракты для DEX и хуков.

2.4.2.3. Chainlink

Оракулы для ценовых данных.

2.4.2.4. Binance API

API для получения исторических данных о ценах.

2.4.2.5. OpenZeppelin Contracts & UniswapHooks

Библиотеки шаблонов.

2.4.2.6. Flask

Фреймворк для веб-интерфейса.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.01-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

[illegible]